CATS NEWSLETTER

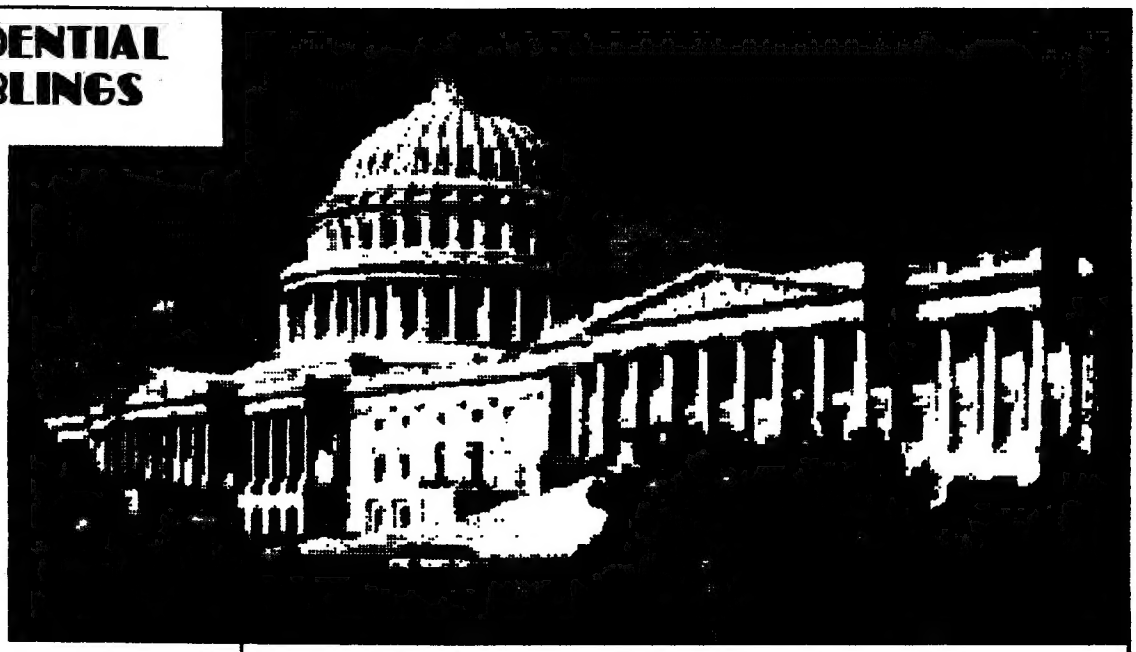
CAPITAL AREA TIMEX SINCLAIR USERS GROUP

P.O. Box 467, Fairfax Station, VA 22039

Volume 7, Number 2

June, 1989

PRESIDENTIAL RAMBLINGS



- **Capital Fest**

We had a big turnout - over 200! I am pleased to note that we had attendees, vendors and presenters from all over the country and also many from our neighbor to the north, Canada! We were pleased that they were able to come. IT WAS SUPER!

I would be remiss if I did not thank the talented members of CATS who put in an enormous amount of time to make the Fest a success. The following people made contributions under the very able leadership of Audrey Curnutt (Fest committee chairman): Tom Bent, Bob Curnutt (Fest committee secretary), Hank Dickson, Mark Fisher and Steve Greene (BBS), Stan Guttenberg (Fest committee treasurer), Joe Miller, Ted Osheroff, Vernon Smith and Mike Warmick (Fest videographer). I would also like to thank those who helped out on the day of the Fest. I would especially like to thank Audrey for her efforts and expertise because she pulled us all together and whipped us into shape!

- **Meeting Notes**

Please remember that there are some important items on the June meeting's agenda - constitutional changes to add a new, elected position of Corresponding Secretary and our annual election of officers. Brief write-ups follow:

- **Nominating Committee Report**

The Executive Board served as the nominating committee of the whole. The following is a report of the names that the committee is placing in nomination for the fiscal year July 1, 1989 through June 30, 1990.

President Bill Barnhart *(Incumbent)*

1st Vice President Hank Dickson *(Incumbent)*

2nd Vice President George Rey *(Incumbent)*

Recording Secretary Bob Curnutt

Corresponding Secretary Joe Miller

Treasurer OPEN

The elections will be held at the June meeting. Nominations can be made from the floor at the meeting or by contacting a Board member prior to the meeting. As you can see from the above list, it is important that we find an individual who is interested in being the Treasurer.

- **New Corresponding Secretary Position**

At the April meeting, the new position of corresponding secretary was discussed and unanimous approval was given to finalize a draft of the necessary Constitution changes. The draft of the changes which will be voted on at the June meeting can be found in your May newsletter.

See you at the meeting on the 10th,

Bill

In This Issue

JUNE MEETING AGENDA

11:00 ————

1:30 **General Meeting**

2:00 **Program:**

Pascal on T/S computers

We plan to present an introduction to Pascal programming in which you will actually write a program, compile it, and run it. With several networked QL's, it should be different than any workshop so far. To be lead by Duane Parker, assisted by Richard Roseen. (Those who agreed to help, please bring your QL's, Trump cards, eprom boards, network cables, and monitors.) As another favor, please bring a QL eprom board, if you have one, and a small desk lamp or clip-on lamp (this is so you can see the keyboard when the room is darkened for projected slides). Come early if you wish to see how to set up a network.

FROM THE EDITOR

What a blast! I don't know about you, but I thoroughly enjoyed myself at the Fest. It was a tributed to the organizing committe that everything went off without a hitch—at leas t that was the outward appearance. I know that Audrey Curnutt was probably a bundle of knots internally, till the Fest closed its doors on Sunday.

FOR THEIR OUTSTANDING JOB, THIS ISSUE OF THE NEWSLETTER IS DEDICATED TO:

THE CAPITAL FEST COMMITTEE AND IT'S CHAIRWOMAN, **AUDREY CURNUTT.**

MANY THANKS.

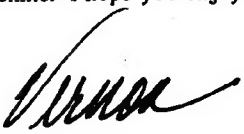
I can't wait to see the video tapes of the presentations. At CATS we are lucky to have the services of Mike Warmick to videotape our meetings and now the Fest proceedings.

Several people remarked that they were disappointed that the attendance wasn't higher. They seemed to feel that our hall ought to resemble the Washington Convention Center during FOSE (Federal Office Systems Expo). Sure, that would have been welcomed but one has to deal in reality. Reality ios that the plug was pulled on the 2068 over 5 years ago. Likewise, the sale of the QL (and everything else Sinclair) was about 3 years back. I think it is phenomenal that we had the showing that we did. We had the "hard core" Sinclair types, and that's what you expect to show up at these events. Even though we are in a major metropolitan area, we know that we have most of the "committed types" in our membership already. Our out of town visitors were the "glue" that holds most of the groups together because if you look, there are usually only a few enthusiasm is sufficient to keep everyone else interested. We reached these people.

Please scan our ad section, The Classifieds, on the next to last page. I know that you're running out of steam by then, but I think it will be worth your time. WE HAVE PAID ADS! How's that for a switch? RMG is still supporting the Sinclair community, so if you need something Sinclair, check with RMG to see if they stock it.

If you're browsing through the June 1989 issue of the National Geographic, check out page 707 which shows a ZX Spectrum 128 in use.

This issue continues Ray Byler's 2068 ROM disassembly and Tim Swenson's banner program. Both were started in previous issues. Duane Parker has prepared an interesting artical on QL Pascal to compliment his meeting presentation. Finally, John Riley makes a confession about his use of an IBM machine. I hope you enjoy the issue.



Submissions to the newsletter should conform to the following criteria: If hard copy is submitted, it must be no wider that 2 3/8" in width. Hard copy is the preferred media for TS 1000 and 2068 based articles. QL users can submit microdrive cartridges or floppy disks. NO HARD COPY. If possible, though, use the BBS, (301) 588-0579. Send to:

CATS

P.O. Box 467

Fairfax Station, VA 22039

NEWS, NEWS, NEWS

Deep Modem tells what's going on

DUANE PARKER is cooking up something special for the next CATS meeting!! The topic is: "Programing with PASCAL". He intends to connect about six QL's on-site in a small network which will permit the attendees to carry out—in real time—the PASCAL lesson he will conduct. He needs a couple of more volunteers to bring QL's and monitors. But he doesn't need just any QL's, rather QL's which have access to the TOOLKIT package. TOOLKIT can be located on an interface card or a multiple ROM card. (Ed note: Many of the EPROMs that Tom Bent has burned also have TOOLKIT.) More commonly, it can be found on a TRUMP card. Other items needed will be QL interface cable and extra extension cords and power strips.

Duane will begin the PASCAL workshop between 1 and 1:30 p.m. Saturday, June 10, 1989. After about an hour, there will be a break for an important CATS business meeting, which will include election of officers. Programing in PASCAL will then resume, concluding about 4:30 p.m.

Although no special hardware events have been scheduled because of the planned absence of Tom Bent, Duane Parker's presentation has enough hardware challenges associated with it to easily take up the slack. Plan to be there, to help, and to learn about PASCAL!!

Who was that bearded man at the CapitalFest? None other than our esteemed Tape Librarian from the Deep South, John Riley! John's been into a lot of things lately. If you've read past issues of the newsletter—I'm assuming that someone has—you know John has a 2068 with a Larken system, besides his QL. John's contribution this month reveals that he's been toiling in other vineyards, as well. The IBM (Boo, hiss) ones to be exact. Before we all come down on him, rwead his article. He makes some very good points. I certainly can't cast any stones. Deep Modem says that sooner or later many of us will b e faced with the same decision that John had to make. THINK ABOUT IT.

Mr. Deep would like to see more of you getting with the program, Telecommunications, that is. What have you got against our BBS. It's sort of a nice way to keep up with things and you can do your newsletter editor a service at ther same time. The lead article on this page was put on the board by that notorious MS-DOS user, Hank Dickson, and it was downloaded to the newsletter. (Now Hank will never have to set up his QL!) No sweat now for Hank and it could be the same for the rest of you. Remember, its "Keep the N/L Editor Happy Time"! Do your part and at the same time expand you knowledge of one of the most interesting areas of personal computer use.

QL BANNER

by Tim Swenson

Originally printed in Timelinez, March, 1987

Continued from the last issue.

```
240     f_key
250     IF key<>0 THEN GO TO 160
260     LET start_at = 32-y
270     IF y>32 THEN start_at=0
280     LET x$=a$(x)&" "
290     AT 4,start_at : PRINT
x$(start_s TO last_s)
300     LET last_s = last_s +1
310     IF y>31 THEN LET start_s
= start_s +1
320   NEXT y
330   AT 4,0: PRINT "  "
340 NEXT x
350 END REPeat loop
360 DEFine PROCedure f_key
370   SELect ON key
380     ON key = 2
390       set_screen2
400       get_string
410       set_screen1
420     ON key = 8
430       clear_data
440       set_screen2
450       get_all
460     ON key = 16
470       set_screen2
480       EXIT loop
490       set_screen1
500       STOP
510   END SELect
520 END DEFine f_key
530 DEFine PROCedure set_screen1
540   MODE 8 : WINDOW 512,256,0,0
550   PAPER 0 : INK 4 : CLS
560   CSIZE 3,1
570   STRIP 1 : AT 2,0: PRINT "
"
580   STRIP 1 : AT 6,0: PRINT "
"
590   STRIP 4 : AT 1,0: PRINT "
"
600   STRIP 4 : AT 7,0: PRINT "
"
610   STRIP 3 : AT 0,0: PRINT "
"
620   STRIP 3 : AT 8,0: PRINT "
"
630    STRIP 0
640 END DEFine set_screen1
650 DEFine PROCedure set_screen2
660   MODE 4 : CSIZE 0,0
670   CLS
680 END DEFine set_screen2
690 DEFine PROCedure clear_data
700   FOR x = 1 TO 10
710     a$(x) = " "
720   NEXT x
730 END DEFine clear_data
740 DEFine PROCedure get_string
750     PRINT "Enter String Number
to Re-enter "
760     INPUT x
770     PRINT : PRINT "Enter New
String Message "
```

QL PASCAL

by Duane Parker

What is Pascal? It is the name of a French mathematician and philosopher (Blaise, P., 1623-1662), but it is also the name of a high-level computer language. This language was developed in 1968 for the main purpose of teaching computer programming. Since that time it has become quite popular as an all-purpose language for use on microcomputers. It is a compiled language - that is, programs are first written as a text file, and then converted to a program that will run. The conversion is performed by a special program, called a "compiler". Usually run-time procedures must be also be linked with the compiled code-file to complete the process. The final result is a machine code program, that can loaded and run, on the QL, by typing in the Qdos command "exec <drive & filename>". Pascal compiler's are available for all the T/S computers. Three of them have been written for the QL!

So, another language!? Isn't BASIC and Archive enough? Why bother with another one? I can offer several reasons: 1) one can use Pascal to write machine code programs for the QL without using assembler or other lower-level languages, 2)learn a widely used standard computer language to improve one's skills, 3)write saleable computer programs that can be compiled on the QL(or even on other computers), and/or 4)obtain Pascal programs written to run on other computers and compile these to run on the QL to accomplish a desired task. OK, suppose you decide to delve into Pascal: what does it look like? First of all, Pascal has commands or statements (like BASIC), symbols that separate those statements, functions (SIN, SQRT, etc., like BASIC) and operators (+,-,*,/, etc., like BASIC), but it does not have line numbers. The primary statements that input and output (from keyboard or files) are: READLN (like INPUT) WRITELN (like PRINT) The assignment statement ":=" resembles the BASIC "LET x=50", for example: count := 50; tax := cost*0.05; name := 'Tom'; (The semi-colon is not part of these statements, it just shows where they end to separate them from the next statement - more on that later). Then there are control statements: IF .. THEN .. ELSE FOR .. TO .. DO (like the FOR/NEXT loop) REPEAT .. UNTIL .. etc. Unlike BASIC, each program must have the statement "PROGRAM & name" at the beginning to show where it starts, and the separator "END." to show where it ends. Here, the period is the most important, in fact, it may work by itself! (I really don't know - all programs that I have seen, or written, have the word END at the end, before the period). The language uses curly brackets "{,}" or parentheses plus an asterisk "(*,*)" to enclose programmer's remarks (instead of the REM statement in BASIC).

Have you followed me this far? Even if you haven't, a look at the appearance of a Pascal program might make things clearer. Here's a very simple sample:

```
PROGRAM hi(INPUT, OUTPUT);

BEGIN (Main part) WRITELN('Hi, CATS!'); WRITELN('Let us look at Pascal.'); (*Prog. is called "hi"*) END.
```

Once compiled and run, this program simply prints:

Hi, CATS! Let us look at Pascal.

Let's look at the structure above. As noted earlier, the program must have a heading (with the statement "PROGRAM name"), each statement must be completed with the semicolon separator. How about "BEGIN"? Begin is not a statment, but a separator; it's used to show where the action starts and show where multiple statements (compound) start. END is also used within the program to show the end of the compound statement and the end of other structures. Lastly, the "INPUT, OUTPUT" in the first line shows that a program may have input and output (Some versions of Pascal always require this.)

Let's try another example:

```
PROGRAM total(INPUT, OUTPUT);

VAR (Variables must be declared, w/ type before use) numb,total:INTEGER;

PROCEDURE showerr; (Procedures establish new commands)
BEGIN WRITELN('Number is too large.'); END;

PROCEDURE add;

VAR (Local to the procd'r) index:INTEGER;

BEGIN FOR index:= 1 to numb DO total := total + index; END;

BEGIN (Main part) READLN(numb); (requires keybd input) IF numb > 200 THEN showerr ELSE BEGIN (Compound statmt)

add; WRITELN('The sum of the first',numb,'integers is: ',total); END; (of compound) END. (of program)
```

Here we have a bunch of lines that look more typical of a Pascal program. Note that the variables, "VAR", must be listed, i. e.: declared, before use. Next, procedures can be defined, as subroutines, and then used in the program to cause an action. Maybe you have been able to guess that this program will print the sum of integers up to what is input, "numb" unless "numb" is above 200.

I have about reached my limit in a short article, - I can't explain all of the structure of Pascal! It does have an exact structure; the programmer is lead by the language to create statements (program code) that are more easily readable, or structured.

I have the QL Pascal compiler, published by Metacomco. What are the mechanics of its use? It came on two MDV cartridges and an eprom that plugs into the slot in the back of my QL. It runs on an unexpanded QL, but it must then use some of the display memory to do all of its work. When the compiler runs, some garbage appears on the screen (more memory prevents this). It comes in three "exec'able" programs: ed, pascal, and paslink.

Why three? First of all one must make up a program on an MDV or disk file. The the screen editor, "ed", is a tool for doing that. To start, the QDOS command "EXEC_W flp1_ed" is used (or "EXEC_W MDV1_ed", flp1_ refers to disk no. 1). This loads and starts the editor, and a menu appears on the screen. Let's say we want to write a small program. First, we

Continued on Page 5

QL Pascal-Continued from Page 4

type in a filename, e. g.: flp1_hi_pas (the "_pas" is not absolutely required, but it shows the type, and the compiler recognizes it.) The next two menu items (Workspace?, and Alter window, [Y/N]?) can be answered with defaults, "ENTER", twice. A nearly blank screen appears, ready for typing (unless flp1_hi_pas already is on flp1 - then the file's contents will appear on screen, ready to be changed). The user then types in the program lines. Ed is a mini-wordprocessor; using it resembles QUILL. Many of the immediate commands are the same: arrows move the cursor around; CTRL-RIGHT, deletes the right character; CRTL-ALT-LEFT, deletes a line; and F3, switches to the command mode. Commands are different than QUILL's, but they do show on one line at the bottom of the window. Thus, when all the typing and editing is finished, F3 & "x" will cause the lines to be written to the file, "flp1_hi_pas", in this case, and exit from the editor occurs.

Now comes the next step, compiling (this is the only program of the three that requires the eprom!). The compiler loads and starts upon entering"EXEC_W flp1_pascal". Six items will come up on the menu, in turn, first: Input source file: type in: flp1_hi (the "_pas" is automatically added). Next: Listing file? (not needed unless a detailed listing of the program is desired), just press "ENTER". Third, Code file? enter: flp1_hi (the binary code will be written to this file). The next three items preform special functions applicable to complex programs; pressing "ENTER" at each prompt sets defaults. Then the compiler reads each line of in "flp1_hi_pas", and converts it into binary code (or object code). Most often the first trial fails because of one or several errors made by the programmer; in this case, a message appears, saying "error no., at line x" and it prints the faulty line. Finally, there is joy in CATSville when the report: "Compilation complete; Any more files to compile [Y/N]?" shows up. Here "N" returns us to QDOS.

Will "hi" now execute? Nope, not yet; the Pascal run-time library must be linked to the binary code file. Entering "EXEC_W flp1_paslink", loads and starts this utility. A menu appears asking: Binary File?, we enter: "flp1_hi", the question is repeated (one can link several binary files with one library), with just one, we type "ENTER". Next it asks: Output File?, we type in "flp1_hi", and "ENTER". Then comes: "Stacksize?" - the default "ENTER" works here. The linker then runs and we have our program "hi". Finally, entering "EXEC flp1_hi" will cause our program to load and run.

In outline form, this is how we generate a Pascal program that runs on the QL. All of the details of Pascal programming just won't fit in this short space. To learn more, and try your hand at the process,

A 2068 Program

by Barry Washington

```
10 REM *** By L.H.WASHINGTON 5/22/89 ***
20 REM *** FOR JLO DISK AND CENTRONICS INTERFACE SYSTEMS WITH EPSON COMPATABLE PRINTERS ***
30 REM *** This Program prints a condensed Catalog listing in a three column format on a (8 1/2 X 11 page size)
40 REM ************************
50 LET /P=O: BORDER 5: PAPER 6: INK 9: CLS
60 PRINT AT 8,6:"DISK CATALOG PRINTER"
70 PRINT AT 13,6:"DO YOU WISH TO VIEW"," ( )CATALOG LIST(Y or N)"
80 INPUT Z$ 90 IF Z$="Y" OR Z$="y" THEN GOSUB 250
100 INPUT "SELECT PRINT COL. 1, 2, OR 3 ";C
110 IF C=1 THEN LET A=0
120 IF C=2 THEN LET A=46
130 IF C=3 THEN LET A=92
140 CLS
150 OUT 127,27: OUT 127,15
160 OUT 127,27: OUT 127,51: OUT 127,25
170 OUT 127,27: OUT 127,69
180 OUT 127,27: OUT 127,108: OUT 127,A
190 OPEN #2,"P"
200 CAT
210 CLOSE #2
220 GO TO 50
230 STOP
240 SAVE /"CAT.PRT" LINE 50
250 CAT: RETURN
```

ADVENTURES IN HERESY

A DIE-HARD T/S USER ENTERS THE WORLD OF MS-DOS

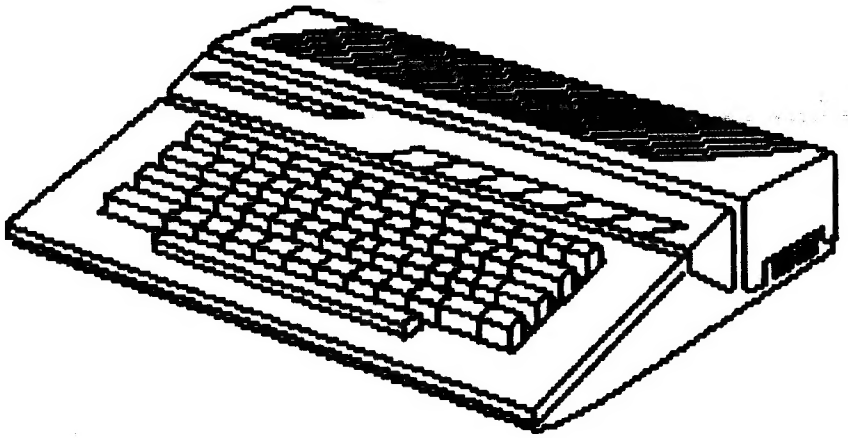
by John Riley

This is an odd column to write. Indeed, I am not even sure that our editor will want to publish it! But the fact must be faced that the "mainstream" of personal computing has flowed onward in the last six years, and certain applications are either beyond the reach of Sinclair machines, or there is no one left who will provide the professional-quality programs or hardware options that we "end-users" want. I realize that the foregoing sentence may be "fightin' words", but it also happens to be the truth. Here is my confession in a nutshell: I have built an XT clone to use as my office computer, replacing the QL after two years of faithful service.

My motives were multiple. First, my church had purchased an AT-compatible machine to do desktop publishing, and I wanted to be able to work in my office on a compatible machine. Second, there is no "computerized Bible" for the QL or 2068, a research tool that I sorely need as a minister. Thirdly, there is a whole world of MS-DOS software out there in Public Domain and Shareware, inexpensive and available (including, by the way, several computerized Bibles). And fourthly, it is now affordable to own an IBM-compatible machine. After all, in 1984 when I bought my 2068 for $175, an XT wasn't even available. IBM was hawking the PC Junior for home use at a vastly inflated price, which was really their PC with a bullet through the spleen. Today, five years later, I can buy a back-up 2068 for $50 (when I can find one), and I can BUILD an XT clone with 512K for about $350, or even less if I scrounge. THIS MEANS THAT IN TERMS OF "DOLLARS PER K OF MEMORY", THE 2068 AND A HOME-BUILT XT COST THE SAME. And remember, this is an unexpanded 2068 that I am talking about. EXPANDING A 2068 FOR DISK DRIVES MIGHT ACTUALLY MAKE IT A MORE EXPENSIVE MACHINE "PER K PER DOLLAR" THAN A HOME-BUILT XT. An interesting observation, don't you think?

"Affordable computing" has always been the watchword of Sinclair enthusiasts. Now, after 5 years, MS-DOS has come into our range. It took them long enough, didn't it?

My original battle plan was to use as many components as possible from my QL system in the construction of the XT. I intended to remove the DRAM from the QL, re-use the Mitsubishi 4853 drives, and rewire my Skip Fisher monitor. It turned out that the DRAMs were soldered into the QL (and I wasn't about to try to desolder them) and that MS-DOS won't recognize 5 1/4" 720K drives (MS-DOS stands for "MicroSoft's Dunderheaded Ornery System"). Those of you with spare 360K drives would have no problem using them. I was able to modify the monitor without any trouble.



The XT components went together pretty easily. I've run into more trouble assembling my kids' toys at Christmas. Anybody who has been through the resin-smoke adventures of the CATS hardware sessions should have no problem building an XT clone. I still haven't gotten it to print to my Seikosha 1000 serial printer, but I'll get that figured out sometime soon. In the meantime I just carry my data disk over to my secretary's office and print stuff out on the laser printer. Nice!

Why, do you ask, didn't I buy the new MS-DOS emulator for the QL? The answers are two — speed and hardware compatibility. The emulator runs at 1 Mhz, my clone runs at 10 Mhz. For twice the investment I got ten times the speed. But just as importantly, I have 6 empty slots on the clone motherboard that I can use to expand the machine in dozens of ways that the QL is incapable of matching. My advice is to reject the emulator as a poor option.

Thus I exit from being a heavy QL-user, but my love affair with Sinclair computers is far from over. For you see, this column was written on my good old 2068, whose keytops may be worn but is still seeing daily use as my family's home computer. It has repaid my investment in it many times over, and I'm sure that it will continue to do so for many years to come. For me and for thousands of others, the orphan Sinclair products are all the computer we need right now for many applications, and in the future.....well, rumor has it the the Z89 will be PC-compatible!

EDITOR'S NOTE: Yes, I know the graphic is an Atari, but it was the best I could do.

2068 ROM DISASSEMBLY by Ray Byler

Continued from the February/March 1989 issue

SPECTRUM ROM ENTRY POINTS INDEXED BY ADDRESS

RESTART ROUTINES & TABLES

HEX	DEC	SPECTRUM NAME	HEX	DEC	TS2068 NAME
0000	00000	START	0000	00000	PLUGIN
0008	00008	ERROR-1	0008	00008	(Print Error)
0010	00016	PRINT-A-1	0010	00016	WRCH
0018	00024	GET-CHAR	0018	00024	(Get Character)
001C	00028	TEST-CHAR	001C	00028	(Tst Character)
0020	00032	NEXT-CHAR	0020	00032	(Get Nxt Char)
0028	00040	FP-CALC	0028	00040	(FP Calculator)
0030	00048	BC-SPACES	0030	00048	(BC Workspaces)
0038	00056	MASK-INT	0038	00056	(Maskable Int)
0048	00072	KEY-INT	0048	00072	(Keyboard Int)
004F	00079	(Pop HL & AF)	004F	00079	PHLAF
0053	00083	ERROR-2	0053	00083	(Error-2)
0055	00085	ERROR-3	0055	00085	LE3
0066	00102	RESET	0066	00102	(NMI Ext Int)
0074	00116	CH-ADD+1	0074	00116	NEXTCH
0077	00119	TEMP-PTR1	0077	00119	NC_HL
0078	00120	TEMP-PTR2	0078	00120	TC_HL
007D	00125	SKIP-OVER	007D	00125	(Control Chrs)
0095	00149	(Token Table)	0098	00152	TOKENS
0205	00517	(Key Tables)	0227	00551	KSCAN
022C	00556	(Ex Mode Ltrs)	0268	00616	(Ex Mode Ltrs)

KEYBOARD ROUTINES

HEX	DEC	SPECTRUM NAME	HEX	DEC	TS2068 NAME
028E	00654	KEY-SCAN	02B0	00688	K_SCAN
0296	00662	KEY-LINE	02B8	00696	(Scanning Loop)
02BF	00703	KEYBOARD	02E1	00737	UPD_K
02F1	00753	K-NEW	0317	00791	(New Key)
0310	00784	K-REPEAT	0336	00822	(Key Repeat Fn)
031E	00798	K-TEST	035C	00860	K_BASE
0333	00819	K-DECODE	0371	00881	CHCODE

LOUDSPEAKER ROUTINES

HEX	DEC	SPECTRUM NAME	HEX	DEC	TS2068 NAME
0385	00949	BEEPER	03F3	01011	PARP
03F8	01016	BEEP	0436	01078	BEEP
046C	01132	REPORT-B	04AA	01194	(Report B)
046E	01134	(Tone Table)	04AC	01196	(Tone Table)

CASSETTE HANDLING ROUTINES

HEX	DEC	SPECTRUM NAME	HEX	DEC	TS2068 NAME
04C2	01218	SA-BYTES	X0B8	X0184	W_TAPE
053F	01343	SA/LD-RET	X0E5	X0229	W_BORD
0556	01366	LD-BYTES	X0FC	X0252	R_TAPE
05E3	01507	LD-EDGE-2	X189	X0393	RD_BIT
05E7	01511	LD-EDGE-1	X18D	X0397	R_EDGE
0605	01541	SAVE-ETC	X1AB	X0427	SLVM
07CB	01995	VR-CONTROL	X58F	X1423	(Verify Command
0802	02050	LD-BLOCK	X5C6	X1478	(Ld Data Block)
0808	02056	LD-CONTRL	X5CC	X1484	LOAD
0886	02230	ME-CONTRL	X6E5	X1765	MERGE
0970	02416	SA-CONTRL	X851	X2129	SAVE
09A1	02465	(Cassette Msgs)	3C89	15497	SEPRMT
09C1	02497	(Program: msg)	3CA9	15529	LDMES

SCREEN & PRINTER HANDLING ROUTINES

09F4	02548	PRINT-OUT	0500	01280	SENDTV
0A11	02577	(Ctrl Char Tbl)	0528	01320	(Ctrl Char Tbl)
0A23	02595	PO-BACK1	053A	01338	P_LFT
0A3D	02621	PO-RIGHT	0554	01364	P_RT
0A4F	02639	PO-ENTER	0566	01382	P_NL
0A5F	02655	PO-COMMA	0576	01398	(Print Comma)
0A69	02665	PO-QUEST	0580	01408	(Print a "?")
0A6D	02669	PO-TV-2	0584	01412	(Ink - Over)
0A9B	02715	(AT Ctrl Char)	0582	01458	SET_AT
0AD9	02777	PO-ABLE	05F0	01520	(Print Chars)
0ADC	02780	PO-STORE	05F3	01523	STTVCU
0AF0	02800	PO-ST-E	0607	01543	(Save Lwr Scrn)
0AFC	02812	PO-ST-PR	0613	01555	(Save Prnt Bfr)
0B03	02819	PO-FETCH	061A	01562	LDTVCU
0B1D	02845	PO-F-PR	0634	01588	(P-Bfr Fetch)
0B24	02852	PO-ANY	0638	01595	(Print Chars)
0B65	02917	PO-CHAR	069A	01690	(Expand Chars)
0B7F	02943	PR-ALL	06B4	01716	(Print a Char)
0B03	03027	PO-ALL-6	0708	01800	(Adjst fr Prtr)
0B08	03035	PO-ATTR	0710	01808	ATTBYT
0C0A	03082	PO-MSG	073F	01855	PUTMES
0C3B	03131	PO-SAVE	0776	01910	PR_TV2
0C41	03137	PO-SEARCH	077C	01916	(Search Table)
0C55	03157	PO-SCR	0790	01936	TVFUL?
0C86	03206	REPORT-5	07C1	01985	ERR5
0CF8	03320	(Scroll? Msg)	0833	02099	(Scroll? Msg)
0D4D	03405	TEMPS	0888	02184	R_ATTS
0D6B	03435	CLS	08A6	02214	K_CLS
0D6E	03438	CLS-LOWER	08A9	02217	CLLHS
0DAF	03503	CL-ALL	08EA	02282	CLS
0DD9	03545	CL-SET	0914	02324	SETCUR
0DD9	03545	CL-SET	0914	02324	SETTVC
0DFE	03582	CL-SC-ALL	0939	02361	SCRL
0E44	03652	CL-LINE	097F	02431	CLS_B

Continued on Page 8

HEX	DEC	SPECTRUM NAME	HEX	DEC	TS2068 NAME
0E88	03720	CL-ATTR	09C3	02499	(Cl Attributes)
0E9B	03739	CL-ADDR	0906	02518	(Get DF Addres)
0EAC	03756	COPY	0A02	02562	K_DUMP
0ECD	03789	COPY-BUFF	0A23	02595	DUMPPR
0EDF	03807	CLEAR-PRB	0A35	02613	CLPR
0EF4	03828	COPY-LINE	0A4A	02634	PRSCAN
0F2C	03884	EDITOR	0A82	02690	EDIT_K
0F81	03969	ADD-CHAR	0AE7	02791	INSA
0FA0	04000	(Edit Keys Tbl)	0B06	02822	(Edit Keys Tbl)
0FA9	04009	ED-EDIT	0B0F	02831	(Do Edit)
0FF3	04083	ED-DOWN	0B59	02905	(Cursor Down)
1007	04103	ED-LEFT	0B6D	02925	(Cursor Left)
100C	04108	ED-RIGHT	0B72	02930	(Cursor Right)
1015	04117	ED-DELETE	0B7B	02939	DELSYM
101E	04126	ED-IGNORE	0B84	02948	(End Edit)
1024	04132	ED-ENTER	0B8A	02954	(Restre ERR-SP)
1031	04145	ED-EDGE	0B97	02967	(Put Cursor)
1059	04185	ED-UP	0BBF	03007	(Cursor Up)
1076	04214	ED-SYMBOL	0BD7	03031	(Sym & Grph Cd)
107F	04223	ED-ERROR	0BE5	03045	(Edit Error)
1097	04247	CLEAR-SP	0BFD	03069	DEL_K
10A8	04264	KEY-INPUT	0C0E	03086	IN_K
111D	04381	ED-COPY	0C83	03203	ECHO
1190	04496	SET-HL	0CF6	03318	(Loc Wrk Space)
11A7	04519	REMOVE-FP	0D0D	03341	DESLUG

EXECUTIVE ROUTINES

HEX	DEC	SPECTRUM NAME	HEX	DEC	TS2068 NAME
11B7	04535	NEW	0D1D	03357	K_NEW
11CB	04555	START/NEW	0D31	03377	INIT
11DA	04570	RAM-CHECK	0D40	03392	(Check Memory)
1219	04633	RAM-SET	0D7F	03455	NEW
12A2	04770	MAIN-EXEC	0E28	03624	(Edit Mode Lp)
12A9	04777	MAIN-1	0E2F	03631	LED18
1303	04867	MAIN-4	0E80	03725	LED4
1391	05009	(Report Msgs)	0F65	03941	RPTMSG
155D	05469	MAIN-ADD	1158	04440	(Add BASIC Line
15AF	05551	(Init Chan Info	11AA	04522	CHINIT
15C4	05572	REPORT-J	11BF	04543	(Invld I/O Dev)
15C6	05574	(Init Strm Data	11C1	04545	SMINIT
15C9	05577	(Sinclair Logo)	1118	04376	(Timex Logo)
15D4	05588	WAIT-KEY	11CF	04559	RDCH
15D4	05588	WAIT-KEY	X8AA	X2218	AKEY
15E6	05606	INPUT-AD	11E1	04577	INCH
15EF	05615	OUT-CODE	11EA	04586	PUTDIG
15F2	05618	PRINT-A-2	11ED	04589	SENDCH
1601	05633	CHAN-OPEN	1230	04656	SELECT
160E	05646	REPORT-O	1230	04669	ERRO
1615	05653	CHAN-FLAG	1248	04680	SEL_HL
162D	05677	(Chan Code Tbl)	1293	04755	(Channel Flags)
1634	05684	CHAN-K	129A	04762	(Set K Flags)
1642	05698	CHAN-S	12A8	04776	(Set S Flags)
164D	05709	CHAN-P	12B3	04787	(Set P Flags)
1652	05714	ONE-SPACE	12B8	04792	INS1
1655	05717	MAKE-ROOM	12B8	04795	INSERT
1664	05732	POINTERS	12CA	04810	REMGSZ
168F	05775	LINE-ZERO	131E	04894	(Find Line No.)
1695	05781	LINE-NO	1324	04900	GET_LN
169E	05790	RESERVE	1320	04909	LCU2
16B0	05808	SET-MIN	133F	04927	CLEL
16BF	05823	SET-WORK	134E	04942	X_CALC
16C5	05829	SET-STK	1354	04948	RESET
16D4	05844	REC-EDIT	1363	04963	X_T_HL
16DC	05852	INDEXER	136B	04971	SEARCH
16E5	05861	CLOSE	139F	05023	CLOSE
16EB	05867	(Make Strm Dt=0	13A8	05032	RSTSTR
1701	05889	CLOSE-2	13BE	05054	CLCHAN
1716	05910	(Clse Strm Tbl)	1407	05127	(Clse Strm Tbl)
171C	05916	CLOSE-STR	1400	05133	(Close Strm Sub
171E	05918	STR-DATA	140F	05135	(Test Strm No.)
1736	05942	OPEN	142A	05162	OPEN
1750	05981	OPEN-2	1465	05221	OPCHAN
177A	06010	(Opn Strm Tbl)	14C7	05319	(Opn Strm Tbl)
1781	06017	OPEN-K	14CE	05326	(Open K Strm)
1785	06021	OPEN-S	14D2	05330	(Open S Stream)
1789	06025	OPEN-P	14D6	05334	(Open P Stream)
1793	06035	CAT-ETC.	25C8	09672	CAT
1793	06035	CAT-ETC.	25CC	09676	FORMAT
1793	06035	CAT-ETC.	25D0	09680	MOVE
1793	06035	CAT-ETC.	25D4	09684	ERASE
1795	06037	AUTO-LIST	14E1	05345	LIST
17F5	06133	LLIST	1541	05441	K_LLST
17F9	06137	LIST	1545	05445	K_LIST
1855	06229	OUT-LINE	15A1	05537	PUT_SR
1860	06240	(LD D,O)	15AC	05548	LPO
187D	06269	OUT-LINE2	15C9	05577	PUT
1886	06326	NUMBER	1602	05634	(Skip Over No.)
18C1	06337	OUT-FLASH	160D	05645	FLASHA
18E1	06369	OUT-CURS	162D	05677	PR_CUR
190F	06415	LN-FETCH	1658	05723	NEXT_L
191C	06428	LN-STORE	1668	05736	DE_HL
1925	06437	OUT-SP-2	1671	05745	(Prnt Char/Tkn)
192A	06442	OUT-SP-NO	1676	05750	(Add Spaces/No)
1937	06455	OUT-CHAR	1683	05763	(Print Line)
196E	06510	LINE-ADDR	16D6	05846	FIND_L
1980	06528	CP-LINES	16E8	05864	CP_BC
1988	06536	(Fnd Stmt Sub)	16F0	05872	SUBLIN
1988	06539	EACH-STMT	16F3	05875	SUBLN1
1988	06584	NEXT-ONE	1720	05920	RECLEN
19DD	06621	DIFFER	1745	05957	(Dif of Length)
19E5	06629	RECLAIM-1	174D	05965	DEL_DE
19E8	06632	RECLAIM-2	1750	05968	DELREC

Continued on Page 9

HEX	DEC	SPECTRUM NAME	HEX	DEC	TS2068 NAME
19FB	06651	E-LINE-NO	1788	05992	LINENO
1A1B	06683	OUT-NUM-1	1788	06024	PUT_BC
1A28	06696	OUT-NUM-2	1795	06037	PUT_LN
		BASIC LINE & COMMAND INTERPRETATION			
1A48	06728	(Cmd Offsets)	1945	06469	(Cmd Offsets)
1ADF	06879	P-SAVE	19ED	06624	TEMP38
1AE0	06880	P-LOAD	19E1	06625	TEMP39
1B17	06935	LINE-SCAN	1A27	06695	SYNTAX
1B28	06952	STMT-LOOP	1A44	06724	LS4
1B52	06994	SCAN-LOOP	1A95	06805	(Get Cmd Class)
1B6F	07023	SEPARATOR	1AB2	06834	(Chk for Sprtr)
1B76	07030	STMT-RET	1AB9	06841	ENDSTT
1B8A	07050	LINE-RUN	1AD8	06872	EXECUTE
1B9E	07070	LINE-NEW	1AEC	06892	(Fnd Adrs Newln
1BB2	07090	REM	1B00	06912	(Rem Command)
1BB3	07091	LINE-END	1B09	06921	(Ftch Add Nxtln
1BBF	07103	LINE-USE	1B15	06933	(Fnd # Newline)
1BD1	07121	NEXT-LINE	1B27	06951	(Set Nxtln use)
1BEE	07150	CHECK-END	1B44	06980	END?
1BF4	07156	STMT-NEXT	1B4A	06986	ENDTEM
1C01	07169	(Cmd Class Tbl)	1B64	07012	(Cmd Class Tbl)
1C0D	07181	CLASS-03	1B70	07024	(Class 3 Cmds)
1C16	07190	JUMP-C-R	1B79	07033	(Jmp to TADDR)
1C1F	07199	CLASS-01	1B82	07042	TEM1
1C2E	07214	REPORT-2	1B91	07057	ERR2
1C59	07257	VAL-FET-2	1BBC	07100	LT22
1C79	07279	NEXT-2NUM	1BDC	07132	DYADIC
1C82	07298	EXPT-1NUM	1BE5	07141	TEM6
1C8A	07306	REPORT-C	1BED	07149	SYNERR
1C8C	07308	EXPT-EXP	1BEF	07151	TEM10
1CDE	07390	FETCH-NUM	1C49	07241	OPTNO
1CE6	07398	USE-ZERO	1C51	07249	STK_0
1CEE	07406	STOP	1C59	07257	STOP
1CF0	07408	IF	1C5B	07259	(If Command)
1D03	07427	FOR	1C78	07288	FOR
1D86	07558	LOOK-PROG	1D28	07464	SKIP
1DAB	07595	NEXT	1D55	07509	NEXT
1DEC	07660	READ-3	1D96	07574	(Read after 1st
1DED	07661	READ	1D97	07575	READ
1E27	07719	DATA	1E82	07810	DATA
1E42	07746	RESTORE	1E9D	07837	(Restore Comman
1E45	07749	REST-RUN	1ECA	07882	RESTBC
1E4F	07759	RANDOMIZE	1ED4	07892	RAND
1E5F	07775	CONTINUE	1EE4	07908	CONT
1E67	07783	GO-TO	1EF1	07921	JUMP
1E73	07795	GO-TO-2	1EFD	07933	GOTO_2
1E7A	07802	OUT	1F04	07940	(Out Command)
1E80	07808	POKE	1F0A	07946	(Poke Command)
1E94	07828	FIND-INT1	1F1E	07966	FIX_U1
1E99	07833	FIND-INT2	1F23	07971	FIX_U

HEX	DEC	SPECTRUM NAME	HEX	DEC	TS2068 NAME
1E9F	07839	REPORT-B	1F29	07977	ERRB
1EA1	07841	RUN	1F2B	07979	(Run Command)
1EAC	07852	CLEAR	1F36	07990	CLEAR
1EAF	07855	CLEAR-RUN	1F39	07993	CLR_BC
1EED	07917	GO-SUB	1F99	08089	GO_SUB
1F05	07941	TEST-ROOM	1FBB	08123	CHK_SZ
1F15	07957	REPORT-4	1FCF	08143	ERR4
1F23	07971	RETURN	1FD4	08148	RETURN
1F3A	07994	PAUSE	1FEB	08171	PAUSE
1F54	08020	BREAK-KEY	2009	08201	BREAK?
1F60	08032	DEF-FN	201D	08221	DEF
1FC3	08131	UNSTACK-Z	214F	08527	SYNTWO
1FC9	08137	LPRINT	2155	08533	K_LPR
1FCD	08141	PRINT	2159	08537	K_PRIN
1FDF	08159	PRINT-2	217E	08574	P_SEQ
2048	08264	PR-ST-END	21E7	08679	TERM?
2070	08304	STR-ALTER	220F	08719	STRITO
2089	08329	INPUT	222B	08747	INPUT
20C1	08385	IN-ITEM-1	226B	08811	I_SEQ
21D4	08660	REPORT-H	237E	09086	ERRH
21D6	08662	IN-CHAN-K	2380	09088	NOTKB?
21E1	08673	CO-TEMP-1	2388	09099	(Tst fr Clr Cd)
21E2	08674	CO-TEMP-2	238C	09100	GR_COL
21F2	08690	CO-TEMP-3	239C	09116	(Test for Ink)
21FC	08700	CO-TEMP-4	23A6	09126	COLITM
2211	08721	CO-TEMP-5	2388	09147	TV_COL
2234	08756	CO-TEMP-7	23DE	09182	COLOUR
2273	08819	CO-TEMP-C	241D	09245	HIFLSH
2294	08852	BORDER	243E	09278	BORDER
22AA	08874	PIXEL-ADD	2603	09731	SCRMBL
22CB	08907	POINT-SUB	2624	09764	F_PNT
22DC	08924	PLOT	2635	09781	PLOT
22E5	08933	PLOT-SUB	263E	09790	PLOTBC
2307	08967	STK-TO-BC	2660	09824	GET_XY
2314	08980	STK-TO-A	266D	09837	GET_A
2320	08992	CIRCLE	2679	09849	CIRCLE
2382	09090	DRAW	26DB	09947	DRAW
24B7	09399	DRAW-LINE	2810	10256	DRAW_L
24BA	09402	(Compare X&Y)	2813	10259	DRAWLN
		EXPRESSION EVALUATION			
24FB	09467	SCANNING	2854	10324	EXPRN
2530	09520	SYNTAX-Z	2889	10377	INTPT?
2535	09525	S-SCRN$-S	288E	10382	F_SCRN
2580	09600	S-ATTR-S	2807	10455	F_ATTR
25AF	09647	S-U-PLUS	296D	10605	(Scanning Func)
25F8	09720	S-RND	2986	10678	RND
2627	09767	S-PI	29E5	10725	F_PI
2634	09780	S-INKEY$	29F2	10738	F_INKY
26C9	09929	S-LETTER	2A87	10887	(Test Variable)
28AB	10411	FN-SKPOVR	2C69	11369	NXT_HL

Continued on Page 10

HEX	DEC	SPECTRUM NAME	HEX	DEC	TS2068 NAME
2882	10418	LOOK-VARS	2C70	11376	FIND_N
2996	10646	STK-VAR	2D54	11604	GET_EL
2A52	10834	SLICING	2E10	11792	SLICER
2AB2	10930	STK-STO-$	2E70	11888	PSHSTR
2AB6	10934	STK-STORE	2E74	11892	PAEDCB
2AFF	11007	LET	2EB0	11965	LET
2B59	11097	L-NUMERIC	2F17	12055	L_NUM
2BF1	11249	STK-FETCH	2FAF	12207	POPSTR
2C02	11266	DIM	2FC0	12224	DIM
2C88	11400	ALPHANUM	3046	12358	ALNUM?
2C8D	11405	ALPHA	3048	12363	ALPHA?
2C9B	11419	DEC-TO-FP	3059	12377	STKUSN
2D1B	11547	NUMERIC	3009	12505	DIGIT?
2D28	11560	STACK-A	30E6	12518	STK_A
2D2B	11563	STACK-BC	30E9	12521	STK_BC
2D3B	11579	INT-TO-FP	30F9	12537	ININT

ARITHMETIC ROUTINES

2D4F	11599	E-TO-FP	3100	12557	XEY
2D7F	11647	INT-FETCH	3130	12605	LDDE
2D8C	11660	P-INT-STO	314A	12618	STDE_U
2D8E	11662	INT-STORE	314C	12620	STDE_S
2DA2	11682	FP-TO-BC	3160	12640	FP2BC
2DD5	11733	FP-TO-A	3193	12691	FP2A
2DE3	11747	PRINT-FP	31A1	12705	OUTPUT
2F9B	12187	PREP-ADD	335A	13146	SUMS
2FBA	12218	FETCH-TWO	3379	13177	SUMSLD
2FDD	12253	SHIFT-FP	339C	13212	SHIFT
300F	12303	SUBTRACT	33CE	13262	SUB
3014	12308	ADDITION	33D3	13267	ADD
30A9	12457	HL=HL*DE	3468	13416	MULT
30CA	12490	MULTIPLY	3489	13449	TIMES
31AD	12717	REPORT-6	356C	13676	ERR6
31AF	12719	DIVISION	356E	13678	DIVIDE
3214	12820	TRUNCATE	35D3	13779	TRUNC
3297	12951	RE-STACK	3656	13910	FLOAT

FLOATING-POINT CALCULATOR

32C5	12997	STK-ZERO	3684	13956	CALC
335B	13147	CALCULATE	371A	14106	CTRO
33A9	13225	TEST-5-SP	3768	14184	ROOM?
33B4	13236	STACK-NUM	3773	14195	STK_M
33C0	13248	MOVE-FP	377F	14207	RAMNO
3406	13318	LOC-MEM	37C5	14277	ARRAY
3449	13385	SERIES-06-ETC.	3808	14344	(Series Gen Sub
346E	13422	NEGATE	382D	14381	NEGATE
34A5	13477	(In Command)	3864	14436	(In Command)
34AC	13484	(Peek Command)	3868	14443	(Peek Command)
34E9	13545	TEST-ZERO	3904	14596	TESTO
350B	13579	FP-0/1	3926	14630	STBOOL

HEX	DEC	SPECTRUM NAME	HEX	DEC	TS2068 NAME
36A0	13984	N-MOD-M	3AB8	15035	INTDIV
36AF	13999	INT	3ACA	15050	INT
36C4	14020	EXP	3ADF	15071	EXP
3713	14099	LN	3B2E	15150	LN
3783	14211	GET-ARGT	3B9E	15262	ANGLE
37AA	14250	COS	3BC5	15301	COS
37B5	14261	SIN	3BD0	15312	SIN
37DA	14298	TAN	3BF5	15349	TAN
37E2	14306	ATN	3BFD	15357	ATN
3833	14387	ASN	3C4E	15438	ASN
3843	14403	ACS	3C5E	15454	ACS
384A	14410	SQR	3C65	15461	ROOT
3851	14417	TO-POWER	3C6C	15468	TO_THE

SPARE LOCATIONS (FILLED WITH FF)

386E	14446				
.					
.					
3CFF	15615				

CHARACTER SET

3D00	15616	(Char Dot Ptrns	3D00	15616	CH_SET

* The Timex 2068 Technical manual lists:

TSNAME	HEX
DELSYM	087E
NEW	0D82
LDMES	3CA8

H.E. Meppler (Sep 85 CATS Newsletter) lists:

TSNAME	HEX	SPNAME	HEX
DELSYM	087E	(ED-DELETE)	1016
NEW	0D82	(RAM-SET)	1219
INPUT	222B	(INPUT)	208E
CALC	3684	(? STK-ZERO)	3254
LDMES	3CA8	(Program: Msg)	09C1

N.A. Pashtoon (May/June 88 Sinclus News) lists:

TSNAME	HEX	SPNAME	HEX
DEL_K	0BFE	(CLEAR-SP)	1097
LDMES	3CA8	(Program: Msg)	(09C1)
LINENO	1768	(E-LINE-NO)	198F
PAUSE	1FEF	(PAUSE)	1F3A
READ	1D96	(READ-3)	1DEC

THE CLASSIFIEDS

ADVERTISING INFORMATION: CATS will run one free 1/4 page "commercial" ad per one year full ($18) membership. Noncommercial ads may be submitted any time. Publication dates for both types will be determined by the Newsletter Editor.

ADVERTISING RATES: Full page, $25; 1/2 page, $15; 1/4 page, $10; 2" X 2 1/2", $7

The Capital Area Timex Sinclair Users Group

is a not-for-profit group devoted to serving the interests of those who own, use, or are interested in the Timex/Sinclair family of computers.

Executive Board

President	Bill Barnhart
1st Vice President	Hank Dickson
2nd Vice President	George Reynolds
Secretary	Mike Warmick
Treasurer	Ruth Feguay
Members at Large	Mark Fisher
	Phil Russo
	Bob Curnutt
Immediate Past President	Tom Bent
Editor & contact person	Vernon Smith (703) 978-1835

CATS maintains a greatis exchange of newsletters with approximately 30 users across the U.S. Clubs not sending a n/l to us for six months are automatically taken off the list.

Meetings

Monthly meetings are held from 11 AM to 4:30 PM, on the second Saturday of each month, at the New Carrollton Public Library.

Newsletter

Membership cost $18 per year, are good for 12 months, and include all privileges (access to libraries, group buys, etc.). A newsletter subscription only is available for $12 per year.

Networks

Timex SIG on CompuServe: wednesday night, 10 P.M. Eastern time (GO CLUB).

02X BBS: (505) 522-7081 FIDO net 15, node 6. East Coast dial (703) 471-4815 FIDO net 18, node 9.

CATS BBS: (301) 588-0579. 8-N-1. Capitalfest and general information. 24 hours a day.

CATS Newsletter
P.O. Box 467
Fairfax Station, VA 22039

FIRST CLASS MAIL

The next meeting of CATS will be held on:

Saturday, June 10, 1989 NO HARDWARE WORKSHOP
1:00 PASCAL Program 2:30 General Meeting

At: New Carrollton Public Library
7414 Riverdale Road (Hwy 410), New Carrollton, MD

IF YOU ARE NOT A MEMBER OF CATS, THIS IS THE ONLY ISSUE YOU WILL RECEIVE

DUES: $18 per year, per family